Quantum Lego
Quantum Codes and their Enumerators from Tensor Networks

Brad Lackey

Microsoft Quantum

April 22, 2024

arXiv:2010.05960 *arXiv:2109.08158* *arXiv:2210.02756* *arXiv:2308.05152*

Table of Contents

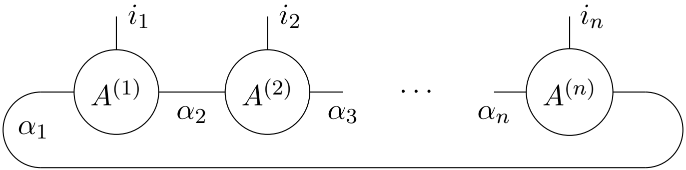
Matrix product states and tensor networks

Tensors are just quantum states $|T\rangle = \sum_{i_1,i_2,\ldots,i_n} T_{i_1 i_2 \ldots i_n} |i_1 i_2 \ldots i_n\rangle$.

- Tensor networks are a way to construct states from lower dimensional ones.
- For example a matrix product state is of the form

$$|T\rangle = \sum_{i_1,i_2,\ldots,i_n} \sum_{\alpha_1,\alpha_2\ldots,\alpha_n} (A^{(1)})^{\alpha_1}_{i_1,\alpha_2} (A^{(2)})^{\alpha_2}_{i_2,\alpha_3} \cdots (A^{(n)})^{\alpha_n}_{i_n,\alpha_1} |i_1 i_2 \ldots i_n\rangle.$$

- Each matrix is viewed as a tensor itself $|A^{(j)}\rangle = \sum_{i_j,\alpha_j,\alpha_{j+1}} (A^{(j)})^{\alpha_j}_{i_j,\alpha_{j+1}} |i_j, \alpha_j, \alpha_{j+1}\rangle$.
- The "α-legs" get traced when forming $|T\rangle$ while the "i-legs" do not.



Quantum codes from tensor networks

For simplicity define a $[[n,k]]_q$ quantum code as a mapping $V : \mathfrak{H}_q^{\otimes k} \to \mathfrak{H}_q^{\otimes n}$, where $\mathfrak{H}_q = \mathbb{C}^q$. One can convert it to a rank $n+k$ tensor $V_{i_1,\ldots,i_{n+k}}$ over a basis

$$V = \sum_{i_j} V_{i_1 \ldots i_{n+k}} |i_{k+1}, \ldots, i_{k+n}\rangle\langle i_1 \ldots, i_k|. \tag{1}$$

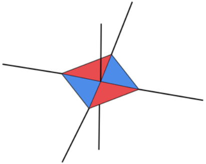
Quantum lego is about building large codes from small.

- Codes are traced like tensors in a network, yielding new codes.
- The check matrix of new code is easily obtained from those of the smaller codes.
- Logical operations are easily analyzed through "operator pushing."

Yet, it is unclear how to obtain the distance of the new code from the smaller ones.

- *Goal*: show the enumerator of the trace is the trace of the enumerators!

A small example: the 2×2 Bacon-Shor code



Today we will work extensively with the 2×2 Bacon-Shor code.

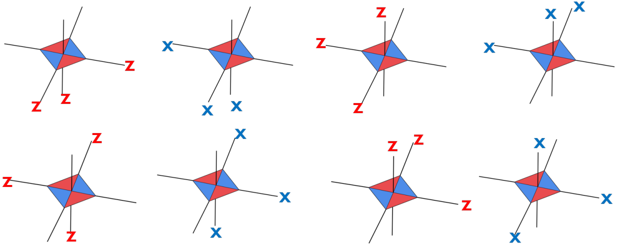
We view it as a $[[4, 2]]$ code with stabilizer $\mathcal{S} = \langle XXXX, ZZZZ \rangle$.

- The four physical qubits are the horizontal legs in the plane.
- The logical qubit is the upward pointing leg.
- The gauge qubit is the downward pointing leg.

Operations on gauge and logical qubits are seen as tensor stabilizers.

For example, on the far left:

$$\overline{Z} = ZIIZ = IZZI$$
$$\overline{X} = XXII = IIXX.$$



Operator pushing

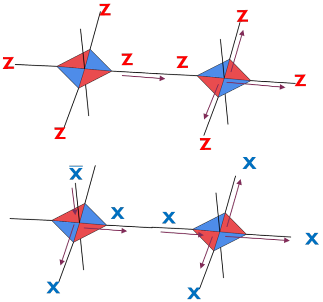
Tracing can be analyzed via “operator pushing.”

Top: $ZZZZZZ$ is stabilizer for the traced code.

- Starting on the left, we push a Z through the trace.
- We use a stabilizer to push this Z to the boundary.

Bottom: we find a representation of a $\overline{X}$ operator.

- We use the local representation of $\overline{X}$.
- We push the X through the trace,
- Finally use a stabilizer to push X to the boundary.



These rules can be formalized into a simple rule for manipulating check matrices.

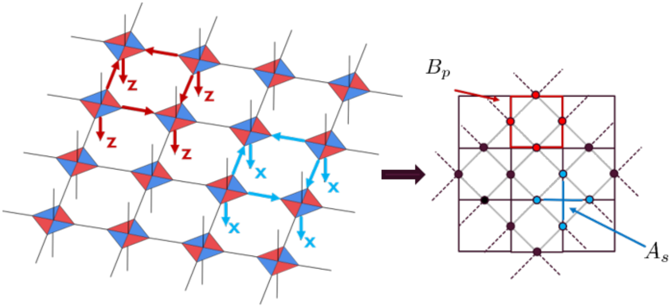
In this example, the traced code is the $[[6, 4, 2]]$ code with $\mathcal{S} = \langle XXXXXX, ZZZZZZ \rangle$.

Constructing topological codes

Here we see a surface code.

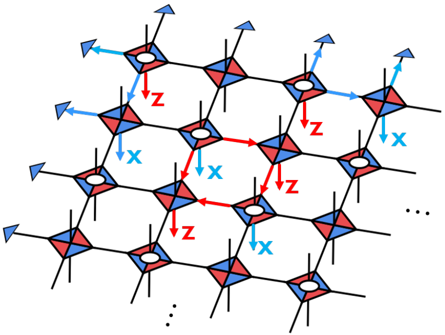
Two sets of gauge operators (pointing downward) each push to the identity.

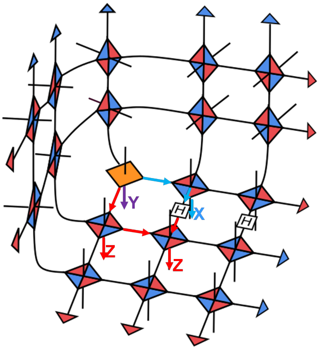
These define relations on these qubits that may be identified as stabilizers of the surface code.



We obtain the toric code by tracing corresponding legs on opposite boundaries.

- Note that the gauge qubits of the legos are now the physical qubits of the code.
- The logical qubits (pointing upwards) have similar relations we have not shown; these form the logical qubits of the code, which are not localized to single legos.





The XZZX-code[1] is recovered with a similar network. Alternating legos have a H applied to the gauge legs.

The triangle code[2] twists the surface code with a [[4, 1]] gadget. Note additional Hs are traced into some legs.

[2]Kay, *PRL* **107**(27):270502, 2011
[2]Yoder and Kim, *Quantum* **1** (2017): 2.

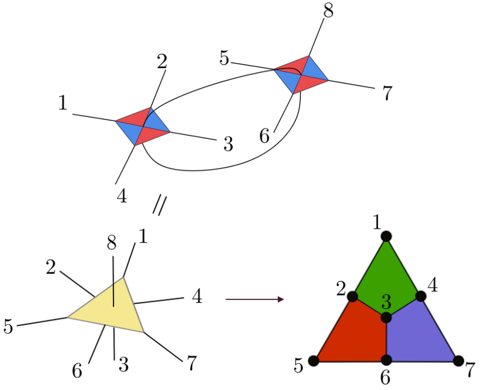
Constructing color codes

Here we build a simple color code: the Steane code.

Again we trace two 2×2 Bacon Shor codes, but now we trace the corresponding logical and gauge qubit legs.

The resulting traced codes is symmetric: we just identify one of the legs as logical (labeled 8 here).

This example illustrates how logical operators and stabilizers intermix in the post-traced code.



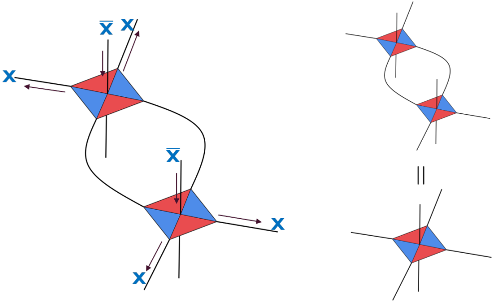
- For example: $XXXXIIII$ is a stabilizer both for one lego and the traced code.
- However, $IIIIXXXX$ is a stabilizer of one lego, but defines a logical operator on the traced code.

Atomic Legos

One can show every quantum code can be constructed from three types of legos:

- A one-qubit state, say $|0\rangle$.
- One qubit unitaries.
- The GHZ state $\frac{1}{\sqrt{2}}(|000\rangle + |111\rangle)$.

Yet, we know of no bounds on how many of these atomic legos will be needed.



For example, in double traced codes shown here, we seem to have a logical operation.

- However these cancel if we push the logical operators on the traced legs.
- Hence $\overline{XX}$ is a relation, and $XXXX$ is a stabilizer of the post-traced code.

In fact, the post-traced code is again the 2×2 Bacon-Shor code.

Quantum weight distributions

Definition [Shor-Laflamme[3]]

Let $\mathcal{E}$ be an error basis on a Hilbert space $\mathfrak{H}$, and M_1, M_2 Hermitian operators on $\mathfrak{H}$ (in practice $M_1 = M_2 = \Pi_{\mathfrak{C}}$). Then their *quantum weight distributions* are

$$A_d(M_1, M_2) = \sum_{E \in \mathcal{E}^n[d]} \operatorname{Tr}(E^\dagger M_1) \operatorname{Tr}(E M_2),$$

$$B_d(M_1, M_2) = \sum_{E \in \mathcal{E}^n[d]} \operatorname{Tr}(E^\dagger M_1 E M_2).$$

Here $\mathcal{E}^n[d] = \{E_1 \otimes \cdots \otimes E_n \; : \; \#\{E_j \neq I\} = d\}$ (e.g. n-qubit Paulis of Hamming weight d).

- B_d is the quantum analogue of the usual weight distribution of a code,
- A_d is the quantum analogue of the weight distribution of its dual code.

[3] Shor and Laflamme, *PRL* **78**(8):1600, 1997.

Quantum weight enumerators

We package up these distributions as polynomials, or enumerators:

$$A(z; M_1, M_2) = \sum_{d=0}^{n} A_d(M_1, M_2) z^d, \quad B(z; M_1, M_2) = \sum_{d=0}^{n} B_d(M_1, M_2) z^d.$$

The "perfect code" is a $[[5, 1, 3]]$ stabilizer code.

- Its stabilizer and some logical operators are shown.
  - There are 16 stabilizers in total.
  - Each logical operator has 16 equivalent representations.

$$\mathcal{S} = \left\langle \begin{array}{l} \texttt{XZZXI}, \\ \texttt{IXZZX}, \\ \texttt{XIXZZ}, \\ \texttt{ZXIXZ} \end{array} \right\rangle$$

Owing to how we have selected to normalize:

$$\begin{aligned} A_{[[5,1,3]]}(z) &= 4 \cdot (1 + 15z^4) \\ B_{[[5,1,3]]}(z) &= 2 \cdot (1 + 30z^3 + 15z^4 + 18z^5). \end{aligned}$$

$$\begin{aligned} X_L &= \texttt{XXXXX} \\ Y_L &= \texttt{YYYYY} \\ Z_L &= \texttt{ZZZZZ} \end{aligned}$$

Two key results about quantum enumerators

Theorem (Rains[4])

Let $U = U_1 \otimes \cdots \otimes U_n$ be a local unitary transformation. Then

$$A(w,z;UM_1U^\dagger, UM_2U^\dagger) = A(w,z;M_1,M_2),$$

and similarly for B.

Theorem (Shor, Laflamme, Rains)

Let $\mathcal{E}$ be an error basis on $\mathfrak{H} = \mathbb{C}^q$ and M_1, M_2 Hermitian operators on $\mathfrak{H}^{\otimes n}$. Then

$$B(w,z;M_1,M_2) = A\left(\frac{w+(q^2-1)z}{q}, \frac{w-z}{q}; M_1, M_2\right).$$

[4] Rains, *IEEE Trans. Inf. Theory* **44**(4), 1998.

Enumerators characterize error detection

For a general $[[n, k, d]]$ stabilizer code $\mathfrak{C}$ we have

- $A_d(\Pi_{\mathcal{C}}, \Pi_{\mathfrak{C}}) = 4^k \cdot \#(\mathcal{S}(\mathfrak{C}) \cap \mathcal{E}^n[d])$.
- $B_d(\Pi_{\mathcal{C}}, \Pi_{\mathfrak{C}}) = 2^k \cdot \#(\mathcal{N}(\mathfrak{C}) \cap \mathcal{E}^n[d])$.
- Hence the distance d is the smallest nonzero power of $\frac{1}{2^k} B(z; \Pi_{\mathfrak{C}}) - \frac{1}{4^k} A(z, \Pi_{\mathfrak{C}})$ with a nonzero coefficient.

Theorem (Ashikhmin, Barg, Knill, Litsyn[5])

Let $\mathfrak{C}$ be a $[[n, k, d]]$-code. Then the probability of an undetected error under a depolarization channel with error probability p is precisely

$$\frac{1}{2^k} B\left(1 - \frac{3p}{4}, \frac{p}{4}\right) - \frac{1}{4^k} A\left(1 - \frac{3p}{4}, \frac{p}{4}\right).$$

[5] Ashikhmin, Barg, Knill, Litsyn, *IEEE Trans. Inf. Theory* **46**(3), 2000.

Tensor enumerators

We create "tensor enumerators" that are tensors with polynomial entries.

- A subset of legs J form the tensor indices (with basis elements $e_{E,E'}$).
- The remaining legs are "traced" into enumerators.
- Here $E \otimes_J F$ means insert factors of E into F at positions in J,

Definition.

Let $\mathcal{E}$ be an error basis on $\mathfrak{H}$, and M_1, M_2 be Hermitian operators. Then for any subset of legs $J \subseteq \{1, \ldots, n\}$ with $|J| = m$, we define:

$$\mathbf{A}^{(J)}(z; M_1, M_2) = \sum_{E,E' \in \mathcal{E}^m} \sum_{F \in \mathcal{E}^{n-m}} \mathrm{Tr}((E \otimes_J F)^\dagger M_1) \, \mathrm{Tr}((E' \otimes_J F) M_2) z^{\mathrm{wt}(F)} e_{E,E'},$$

$$\mathbf{B}^{(J)}(z; M_1, M_2) = \sum_{E,E' \in \mathcal{E}^m} \sum_{F \in \mathcal{E}^{n-m}} \mathrm{Tr}((E \otimes_J F)^\dagger M_1 (E' \otimes_J F) M_2) z^{\mathrm{wt}(F)} e_{E,E'}.$$

Properties of tensor enumerators

Proposition

Let M_1, M_2 and M_1', M_2' be Hermitian operators on $\mathfrak{H}$ and $\mathfrak{H}'$ respectively. Let J and J' be set of legs on these. Then

$$\mathbf{A}^{(J \cup J')}(z; M_1 \otimes M_1', M_2 \otimes M_2') = \mathbf{A}^{(J)}(z; M_1, M_2) \otimes \mathbf{A}^{(J')}(z; M_1', M_2'),$$

and similarly for $\mathbf{B}$.

Theorem (Cao-L.)

Let $\mathcal{E}$ be a unitary error basis on $\mathbb{C}^q$, Hermitian operators M_1, M_2 on $(\mathbb{C}^q)^{\otimes n}$, and $J \subseteq \{1, \ldots, n\}$ with $m = |J|$, then with $\Phi(e_{E,E'}) = \frac{1}{q^{2m}} \sum_{F,F' \in \mathcal{E}^m} \mathrm{Tr}(F^\dagger E F' (E')^\dagger) e_{F,F'}$.

$$\mathbf{B}^{(J)}(w, z; M_1, M_2) = \Phi \left[\mathbf{A}^{(J)} \left(\frac{w + (q^2 - 1)z}{q}, \frac{w - z}{q}; M_1, M_2 \right) \right],$$

Tracing tensor enumerators

The main result can be summarized by the following points.

- The network of the tensor enumerator is the same as that of the underlying code.
- The enumerator of the trace of two legos is the trace of the constituent enumerators (formally at left).

Theorem (Cao-L.)

Suppose $j, k \in J \subseteq \{1, \ldots, n\}$, then

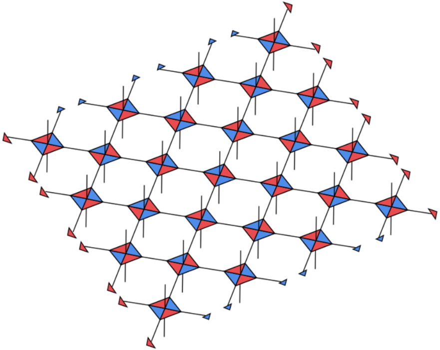
$$\wedge^{j,k}\mathbf{A}^{(J)}(z; M_1, M_2) = \mathbf{A}^{(J\setminus\{j,k\})}(z; \wedge^{j,k}M_1, \wedge^{j,k}M_2),$$

and similarly for $\mathbf{B}^{(J)}$.

- Therefore, if the number of untraced legs does not grow too large we can exactly compute enumerators for quantum codes at scale.

Roughly speaking the complexity is exponential in the size largest cut encountered when tracing the network.

Enumerators of surface codes



For the surface code, at left is a $d = 4$ example.

The lego boundaries have been traced against stabilizer states (and so these are not the physical qubits of the code).

The logical qubits of the legos (downward pointing legs) are the physical qubits of the surface code.

The gauge qubits of the legos (upward pointing legs) are in aggregate the logical qubit.

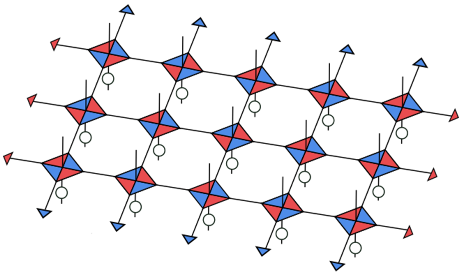
Scaling up, the quantum weight enumerators of a $[[181, 1, 10]]$ surface code are

$$A(z) = 1 + 36z^3 + 180z^4 + 136z^5 + 1344z^6 + 7084z^7 + 24001z^8 + 60432z^9 + 286748z^{10} + \ldots$$
$$B(z) = 1 + 36z^3 + 180z^4 + 136z^5 + 1344z^6 + 7084z^7 + 24001z^8 + 60432z^9 + 286768z^{10} + \ldots$$

The rotated surface code uses the same quantum legos just assembled in slightly different way.

To the left we have an asymmetric 3×5 code.

Similarly, the logical qubits (downward pointing) are the physical qubits of the surface code, while the gauge qubits (upward point) are in aggregate the logical qubit.

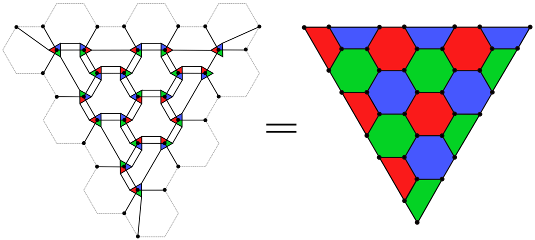


For a $[[256, 1, 16]]$ rotated surface code a

$$
\begin{aligned}
A(z) &= 1 + 30z^2 + 776z^4 + 15538z^6 + 276801z^8 + 4431408z^{10} + 65676619z^{12} \\
&\quad + 912021486z^{14} + 12003931907z^{16} + 150911390280z^{18} + \dots \\
B(z) &= 1 + 30z^2 + 776z^4 + 15538z^6 + 276801z^8 + 4431408z^{10} + 65676619z^{12} \\
&\quad + 912021486z^{14} + 12004980483z^{16} + 150970896992z^{18} + \dots
\end{aligned}
$$

Enumerators of color codes

Here we see the construction of the $d = 7$ color code (from the 6.6.6 family) built from Steane codes as legos.



Steane codes can be built from smaller legos, but are themselvees small enough to be use as basic building blocks

The enumerators for a scaled up $[[91, 1, 11]]$ code from this family are

$$A(z) = 1 + 54z^4 + 297z^6 + 2889z^8 + 24258z^{10} + 197493z^{12} + 1629738z^{14} + 13287999z^{16} + 108647952z^{18} + \dots$$

$$B(z) = 1 + 54z^4 + 297z^6 + 2889z^8 + 24258z^{10} + 4176z^{11} + 197493z^{12} + 67242z^{13} + 1629738z^{14} + 1066740z^{15} + 13287999z^{16} + 14401674z^{17} + 108647952z^{18} + \dots$$

Thank You

Questions:

- Can we find a simple lego constructions for quantum LDPC codes?
- Can we extend these constructions to analyze fault-tolerant circuits?
  - *Answer*: Yes! Stay tuned....
- Can use tensor enumerators to predict how to build good codes using legos?

arXiv:2010.05960 *arXiv:2109.08158* *arXiv:2210.02756* *arXiv:2308.05152*